第二十三节 服务治理

核心 SPI

Dubbo Registry（注册中心）

org.apache.dubbo.registry.RegistryFactory

运行时实现 - org.apache.dubbo.registry.RegistryFactory$Adaptive（动态创建）

比如：

```
registry://127.0.0.1:2181/org.apache.dubbo.registry.RegistryService?application=zookeeper-dubbo-provider&dubbo=2.0.2&environment=product&export=rest%3A%2F%2F192.168.3.22%3A8081%2Forg.geektimes.projects.user.service.EchoService%3Fanyhost%3Dtrue%26application%3Dzookeeper-dubbo-provider%26bind.ip%3D192.168.3.22%26bind.port%3D8081%26deprecated%3Dfalse%26dubbo%3D2.0.2%26dynamic%3Dtrue%26environment%3Dproduct%26generic%3Dfalse%26interface%3Dorg.geektimes.projects.user.service.EchoService%26metadata-type%3Dcomposite%26methods%3Decho%26pid%3D15376%26release%3D2.7.10%26server%3Dnetty%26side%3Dprovider%26timestamp%3D1621515344181&id=REGISTRY#DEFAULT&metadata-type=composite&pid=15376&registry=zookeeper&release=2.7.10&timestamp=1621515281630
```

关键参数：registry=zookeeper

> 注：ServiceConfig 可以使用全局 RegistryConfig，也可以关联自己的 RegistryConfig，影响 Invoke URL 的 registry 的参数
>
> ServiceConfig 配置了服务接口，服务接口生成对应的 Invoker，Invoker 会关联指定 registry 实现

根据 Registry URL 中的 registry 参数找到名称匹配的 org.apache.dubbo.registry.RegistryFactory 的实现，比如 registry=zookeeper，会找到 org.apache.dubbo.registry.zookeeper.ZookeeperRegistryFactory。

创建具体的 Registry 实现，比如：

org.apache.dubbo.registry.zookeeper.ZookeeperRegistryFactory

创建：

org.apache.dubbo.registry.zookeeper.ZookeeperRegistry

org.apache.dubbo.registry.Registry

职责 - 注册与发现

注册

register(URL)

unregister(URL)

发现

subscribe(URL)

unsubscribe(URL)

lookup(URL)

服务暴露调用链路

- ServiceConfig#export()
  - org.apache.dubbo.rpc.Protocol$Adaptive （动态生成代码）
    - org.apache.dubbo.qos.protocol.QosProtocolWrapper#export(Invoker)
      - org.apache.dubbo.rpc.protocol.ProtocolFilterWrapper#export(Invoker)
        - org.apache.dubbo.rpc.protocol.ProtocolListenerWrapper#export(Invoker)
          - org.apache.dubbo.registry.integration.RegistryProtocol#export(Invoker)
            - org.apache.dubbo.registry.Registry#register(URL)

```
rest://192.168.3.22:8081/org.geektimes.projects.user.service.EchoService?anyhost=true&application=zookeeper-dubbo-provider&deprecated=false&dubbo=2.0.2&dynamic=true&environment=product&generic=false&interface=org.geektimes.projects.user.service.EchoService&metadata-type=composite&methods=echo&pid=15376&release=2.7.10&server=netty&side=provider&timestamp=1621515344181
```

ServiceConfig 在服务暴露时，会将当前服务 URL 注册到 Registry 实现上。

ReferenceConfig 在服务消费时，是如何 Registry 交互的?

服务消费调用链路

- org.apache.dubbo.config.ReferenceConfig#get()
  - org.apache.dubbo.config.ReferenceConfig#init()
    - org.apache.dubbo.config.ReferenceConfig#createProxy()
      - org.apache.dubbo.rpc.Protocol$Adaptive#refer(Class,URL)
        - org.apache.dubbo.qos.protocol.QosProtocolWrapper#refer(Class,URL)
        - org.apache.dubbo.rpc.protocol.ProtocolFilterWrapper#refer(Class,URL)
          - org.apache.dubbo.rpc.protocol.ProtocolListenerWrapper#refer(Class,URL)
            - org.apache.dubbo.registry.integration.RegistryProtocol#refer(Class,URL)

ReferenceConfig 组装 Registry URL

```
registry://127.0.0.1:2181/org.apache.dubbo.registry.RegistryService?application=zookeeper-dubbo-consumer&dubbo=2.0.2&environment=product&id=zookeeper&metadata-type=composite&pid=17016&refer=application%3Dzookeeper-dubbo-consumer%26dubbo%3D2.0.2%26environment%3Dproduct%26interface%3Dorg.geektimes.projects.user.service.EchoService%26metadata-type%3Dcomposite%26methods%3Decho%26pid%3D17016%26register.ip%3D192.168.3.22%26release%3D2.7.10%26side%3Dconsumer%26sticky%3Dfalse%26subscribed-services%3Dzookeeper-dubbo-provider%26timestamp%3D1621517180492&registry=zookeeper&release=2.7.10&timestamp=1621517180516
```

registry=zookeeper，同样，来自于 ReferenceConfig 关联的 RegistryConfig。

服务治理组件关联

关联组件	ServiceConfig	ReferenceConfig
Protocol$Adaptive	org.apache.dubbo.rpc.Protocol	org.apache.dubbo.rpc.Protocol
QosProtocolWrapper	org.apache.dubbo.qos.server.Server	org.apache.dubbo.qos.server.Server
ProtocolFilterWrapper	org.apache.dubbo.rpc.Filter	org.apache.dubbo.rpc.Filter
ProtocolListenerWrapper	org.apache.dubbo.rpc.ExporterListener	N/A
RegistryProtocol	org.apache.dubbo.registry.RegistryFactory	org.apache.dubbo.registry.RegistryFactory

Protocol 链路

QosProtocolWrapper -> ProtocolFilterWrapper -> ProtocolListenerWrapper -> RegistryProtocol

传统注册中心实现

抽象实现 - org.apache.dubbo.registry.support.FailbackRegistry

Zookeeper 实现 - org.apache.dubbo.registry.zookeeper.ZookeeperRegistry

以目录结构为代表注册中心

Dubbo 服务提供方法数据存储

- 路径

```
/dubbo/org.geektimes.projects.user.service.EchoService/providers
```

- 路径模式：

```
/${group:dubbo}/${dubbo.service.interface}/${category:providers}/${encode(url.toFullString())}
```

- ZK 数据

```
dubbo%3A%2F%2F192.168.3.22%3A20880%2Forg.geektimes.projects.user.service.EchoService%3Fanyhost%3Dtrue%26application%3Dzookeeper-dubbo-provider%26deprecated%3Dfalse%26dubbo%3D2.0.2%26dynamic%3Dtrue%26environment%3Dproduct%26generic%3Dfalse%26interface%3Dorg.geektimes.projects.user.service.EchoService%26metadata-type%3Dcomposite%26methods%3Decho%26pid%3D2792%26release%3D2.7.10%26side%3Dprovider%26timestamp%3D1621514589203

rest%3A%2F%2F192.168.3.22%3A8081%2Forg.geektimes.projects.user.service.EchoService%3Fanyhost%3Dtrue%26application%3Dzookeeper-dubbo-provider%26deprecated%3Dfalse%26dubbo%3D2.0.2%26dynamic%3Dtrue%26environment%3Dproduct%26generic%3Dfalse%26interface%3Dorg.geektimes.projects.user.service.EchoService%26metadata-type%3Dcomposite%26methods%3Decho%26pid%3D2792%26release%3D2.7.10%26server%3Dnetty%26side%3Dprovider%26timestamp%3D1621514589814
```

- Decode 后的数据：

```
dubbo://192.168.3.22:20880/org.geektimes.projects
.user.service.EchoService?
anyhost=true&application=zookeeper-dubbo-
provider&deprecated=false&dubbo=2.0.2&dynamic=tru
e&environment=product&generic=false&interface=org
.geektimes.projects.user.service.EchoService&meta
data-
type=composite&methods=echo&pid=2792&release=2.7.
10&side=provider&timestamp=1621514589203

rest://192.168.3.22:8081/org.geektimes.projects.u
ser.service.EchoService?
anyhost=true&application=zookeeper-dubbo-
provider&deprecated=false&dubbo=2.0.2&dynamic=tru
e&environment=product&generic=false&interface=org
.geektimes.projects.user.service.EchoService&meta
data-
type=composite&methods=echo&pid=2792&release=2.7.
10&server=netty&side=provider&timestamp=1621514589814
```

Nacos 实现

以数据接口为代表注册中心

服务自省实现（Cloud-Native 实现）

服务订阅

当某个 ReferenceConfig 指定服务接口 XService 时，并且配置了单个注册中心（注册中心实现、IP + 端口），从注册中心拿到 XService 多个 URL

- 调用 org.apache.dubbo.registry.NotifyListener#notify(List)
  - 更新方式
    - 同步更新 - Registry#subscribe 方法
    - 异步更新 - 异步线程调用 org.apache.dubbo.registry.NotifyListener
  - 实现方法 - org.apache.dubbo.registry.integration.RegistryDirectory#notify
    - org.apache.dubbo.registry.integration.RegistryDirectory#refreshOverrideAndInvoker
      - org.apache.dubbo.registry.integration.RegistryDirectory#refreshInvoker

org.apache.dubbo.registry.NotifyListener 实现

Dubbo Cluster（集群）

org.apache.dubbo.rpc.cluster.Cluster

重要抽象实现 - org.apache.dubbo.rpc.cluster.support.AbstractClusterInvoker

关联 org.apache.dubbo.rpc.cluster.Directory 实现，具体参考：org.apache.dubbo.rpc.cluster.support.AbstractClusterInvoker#list

```
    protected List<Invoker<T>> list(Invocation invocation) throws RpcException {
        return directory.list(invocation);
    }
```

当 ReferenceConfig#get() 方法返回某个 Service 代理对象，当代理对象的方法执行时，生成 Invocation 对象（服务接口、服务方法，服务参数），比如：

```
echoService.echo("Hello,World")
```

服务接口：org.geektimes.projects.user.service.EchoService

服务方法：echo

参数类型（列表）：java.lang.String

参数对象（列表）："Hello,World"

已知组件：注册中心（Registry）、服务目录（Directory）、路由器（Router）、负载均衡器（LoadBalance）

Cluster 组件执行链路

ReferenceConfig 配置阶段

- 配置服务接口
- 配置注册中心
- 配置协议（可选）

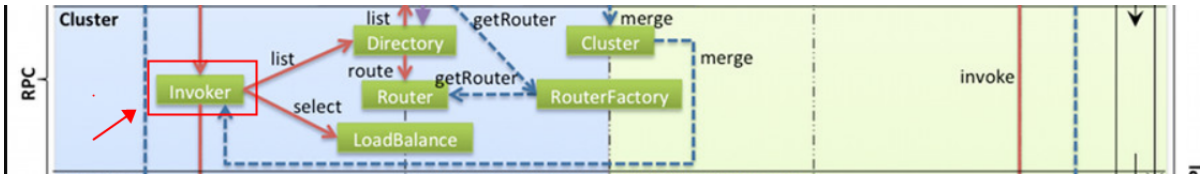
ReferenceConfig 生成代理阶段

ReferenceConfig#createProxy() 方法生成 JDK 动态代理，需要依赖 InvocationHandler 实现，即 org.apache.dubbo.rpc.proxy.InvokerInvocationHandler

ReferenceConfig 代理执行阶段

执行 Invoker （链式）实现，比如：org.apache.dubbo.registry.client.migration.MigrationInvoker -> org.apache.dubbo.rpc.cluster.support.wrapper.MockClusterInvoker，其中 MockClusterInvoker 将调用

org.apache.dubbo.rpc.cluster.interceptor.ClusterInterceptor，从此将 AbstractClusterInvoker 实现关联起来。



其中，AbstractClusterInvoker#list(Invocation) 方法将关联 Directory#list(Invocation) 方法。

通过 Invocation 对象封装，从注册中心（Registry）获取服务提供方的 URL 集合，然后通过服务目录（Directory）转化并存储为 Invoker 集合，再通过路由器（Router）链路由出 Invoker 子集，再经过负载均衡器（LoadBalance）选择其中一个 Invoker 执行 - org.apache.dubbo.rpc.Invoker#invoke(Invocation)。

服务目录 - org.apache.dubbo.rpc.cluster.Directory

存储服务订阅 Invoker 目录（仓库）

静态服务目录 - org.apache.dubbo.rpc.cluster.directory.StaticDirectory

动态服务目录 - org.apache.dubbo.registry.integration.DynamicDirectory

注册中心服务目录实现 - org.apache.dubbo.registry.integration.RegistryDirectory

扩展 org.apache.dubbo.registry.integration.DynamicDirectory，并实现 org.apache.dubbo.registry.NotifyListener

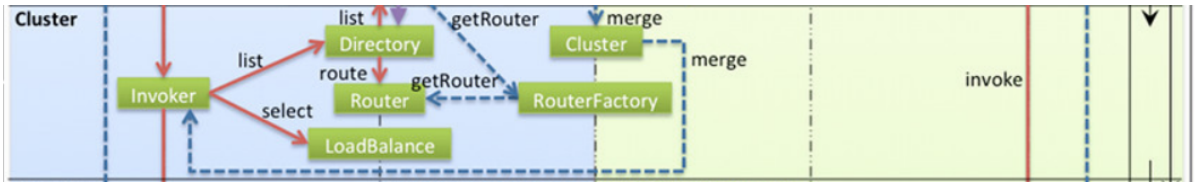
Invoker 路由器 - org.apache.dubbo.rpc.cluster.Router

Router 责任链 - org.apache.dubbo.rpc.cluster.RouterChain

关联多个 Router 实例，并执行路由方法 route:

```
    public List<Invoker<T>> route(URL url,
Invocation invocation)
        // 关联 invokers，假设 invokers 元素有 5 个
        // 经过 routers(3实例)，finalInvokers 变为 2
个
        List<Invoker<T>> finalInvokers =
invokers;
        for (Router router : routers) {
            finalInvokers =
router.route(finalInvokers, url, invocation);
        }
        return finalInvokers;
    }
```

invokers 是服务订阅时，从 RegistryDirectory 将注册中心获取的 URL 集合 转化为的 Invoker 集合，并且关联到 RouterChain，再执行 route 方法，将 Invoker 集合进行路由，产生 Invoker 子集（子集可能与原集合一致）：



org.apache.dubbo.rpc.cluster.LoadBalance

将多个 Invokers（集合）挑选出其中一个，执行在 org.apache.dubbo.rpc.cluster.Router 之后

Dubbo RPC（远程调用）

org.apache.dubbo.rpc.protocol.ProtocolFilterWrapper

org.apache.dubbo.rpc.Filter

问题

1. 为什么暴露的时候是Invoker作为参数，订阅的时候用URL作为参数呢，都用 Invoker不是更好吗

服务消费（订阅）还未生成 Invoker，先去注册中心获取服务提供方服务 URL（集合），再合成 Invoker（集合）

服务暴露（注册）指定的服务接口是确定的，并且在 IP 加端口下生成 Invoker 是确定。Invoker -> Exportor 1对1 关联的。

2. 一个invoker的本质是什么？就是一个接口吗？

一个服务接口暴露时，需要暴露一个服务句柄，代理接口，统一抽象，适配不同的服务接口定义

Invoker 作为代理对象，Invocation 作为调用上下文

org.apache.dubbo.rpc.Invocation

- getMethodName 服务方法名称
- getServiceName 服务接口名称
- getAttachments 相当于元数据，类似于 HTTP Headers